Министерство образования и науки Российской Федерации

Омский государственный университет им. Ф.М. Достоевского

О.Г. Чанышев

Основные элементы
языка программирования Icon

Учебное пособие

Изд-во ОмГУ

Омск 2004

УДК 519.682
ББК 32.973-018я73
Ч 188

Рекомендовано к изданию в качестве учебного пособия учебно-методическим советом математического факультета

Рецензенты:
канд. физ.-мат. наук, доцент *С.В. Зыкин,*
канд. физ.-мат. наук, доцент *А.Л. Агафонов*

Чанышев О.Г.

Ч 188 Основные элементы языка программирования Icon: Учебное пособие. – Омск: Изд-во ОмГУ, 2004. – 55 с.

ISBN 5-7779-0523-4

Пособие является первой публикацией на русском языке, посвященной мощному средству быстрой разработки программ для систем ИИ, особенно для автоматического анализа естественно-языковых текстов – языку Icon. В основу положена книга Thomas W. Christopher «Icon Programming Language Handbook», однако данное пособие – не перевод. Автор опирается на собственный опыт работы с языком; примеры программ в подавляющем большинстве оригинальны.

Для студентов математического факультета.

УДК 519.682
ББК 32.973-018я73

ISBN 5-7779-0523-4

ОГЛАВЛЕНИЕ

Введение 5
1. О синтаксисе, переменные, декларации 11
2. Генераторы
2.1. Every, to, every do 13
2.2. !e – генерация элементов 15
2.3. Оператор & – and 15
2.4. Backtracking и неуспех 15
2.5. Проверка на &null 16
2.6. Альтернатива | 16
2.7. Повторы с ограничением. Функция генерации последовательностей 17
2.8. Coevaluation – «Совычисления» 17
2.9. Множественные арифметические операции сравнения 18
3. Управляющие структуры
3.1. if... 19
3.2. case of 19
3.3. while do 19
3.4. until do 20
3.5. repeat – повторять до бесконечности. 20
4. Символьные множества
4.1. Операции с множествами символов 21
4.2. Принадлежит ли символ множеству? Функция any 21
5. Строки 22
5.1. Подстроки 22
5.2. Операторы для работы со строками 23
5.3. Функции для преобразования строк 23
5.4. Функции сканирования строк 24
5.5. Более мощные варианты функций сканирования 27
6. Списки
6.1. Создание списков 29
6.2. Индексированный доступ к элементам списков 29
6.3. Операции со списками 29
6.4. Функции для работы со списками 30
6.5. Списки как стеки или очереди 30
7. Таблицы 31
7.1. Сортировка таблиц 32
7.3. Некоторые другие функции 33
8. Множества 34
9. Записи 35

10. Процедуры
10.1. Вызов процедур 36
10.2. Выход из процедуры 38
11. Основные функции для работы с файлами 39
12. Немного о совыражениях – coexpression 41
13. Графический интерфейс
13.1. Атрибуты окна 44
13.2. Текст – запись в окно и чтение из 46
13.3. Диалоги 47
13.4. Создание меню 48
13.5. Чтение и запись изображений 50
Заключение 51
Задачи для закрепления материала 52
Список используемой литературы 54

Введение

В предисловии к одной из лучших монографий, посвященных Прологу [1], И. Братко писал: «В средние века знание латинского и греческого языков являлось существенной частью образования любого ученого. Ученый, владеющий только одним языком, неизбежно чувствовал себя неполноценным, поскольку он был лишен той полноты восприятия, которая возникает благодаря возможности посмотреть на мир сразу с двух точек зрения. Таким же неполноценным ощущает себя сегодняшний исследователь в области искусственного интеллекта, если он не обладает основательным знакомством как с Лиспом, так и с Прологом – этими двумя основополагающими языками искусственного интеллекта, без знания которых невозможен более широкий взгляд на предмет исследования».

С полной уверенностью могу добавить к этим языкам и Icon, который прост и могуч во многом потому, что наследует идеологию мощного семейства языков SNOBOL, первые представители которых появились еще в начале 60-х. Название языка не имеет ничего общего с «иконками», а является сокращением от слова «iconoclastic» (иконоборец) [1], используемом в смысле борьбы с конформизмом в разработке языков программирования.

Настоящий материал следует рассматривать как введение в программирование на Icon’е. Это – не учебник по программированию, он рассчитан на студентов, уже имеющих опыт программирования на Си или Паскале. «Императивная» внешность Icon’а способствует быстрому усвоению языка. Простота синтаксиса, небольшое, но функционально мощное множество структур данных делают его очень удобным языком быстрого прототипирования, т. е. за более короткое время, чем программирование на Си, можно написать и отладить программу, при помощи которой проверяются основные идеи концепции.

Сохраняя внешне стиль императивных языков, на самом деле Icon имеет много инструментов, роднящих его с языками логического программирования. Например, «бэктрекинг», или генераторы. Christopher [2] пишет: «Самое большое различие между Icon'ом и другими языками программирования – это то, что выражения Icon'а – генераторы». Выражения генерируют последовательности значений, используя «бэктрекинг».

Пример использования генератора:

```
procedure main()
L:=["mmm","nnnnn","oooooo"] # Список строк
every writes(" ",! L)
end
```

Выход: mmm nnnnn oooooo

Icon не является строго типизированным (тип имеют константы, но не переменные). Это и не плохо и не хорошо, но чаще всего удобно. Будьте внимательны и следуйте модульному стилю программирования.

Icon обладает мощными встроенными типами данных для организации сложно структурированной информации: ассоциативные таблицы, списки, записи, множества по модулю 2 (автоматически исключающие дублирование), символьные множества. Почти любой тип из этого перечня может быть элементом другого типа. Например, элементами таблиц могут быть списки. Списки могут состоять из записей и т. п. Только не надо специально усложнять структуры. Все должно быть ЦЕЛЕСООБРАЗНЫМ. («Красота – это целесообразность» – Иван Ефремов (?)).

Процедуры в Icon'е «относятся к величинам первого класса (first class values, что означает возможность присваивания переменным значений самих процедур, а не результатов их выполнения).

В Icon встроен механизм «совыражений» (co-expression) – почти независимо выполняющихся частей программы, что дает возможность как создавать классические сопрограммы, так и реализовывать произвольные псевдопараллельные алгоритмы.

Icon поддерживает автоматическую конверсию типов.

Icon включает кросс-платформенные графические возможности.

Мой собственный опыт подтверждает, что программы на Icon'е быстро разрабатываются, быстро и надежно работают. Я даже стал задумываться: а не изменить ли моей старой любви – Prolog'у?! (Тем более что Icon распространяется свободно (не для коммерческого использования!) и Вы можете свободно распро-

странять собственные выполняемые модули (exe – файлы), в отличие от PDC Visual Prolog.)

Приведем несколько «программулек» для иллюстрации:

```
procedure main()
#Перезапись файла
f:=open("x.txt","r")|stop("cannot open x.txt")
f2:=open("y.txt","w")|stop("cannot open y.txt")
# Классический стиль
while s:=read(f) do write(f2,s)
close(f)
close(f2)
end
```

В стиле Icon'a подчеркнутые элементы кода заменит выражение

```
every write(f2,!f)
```

```
link graphics
procedure main()
# Простейший вариант открытия графического окна:
WOpen("size=400,300") | stop("can't open window")
Font("Times New Roman,18") # Устанавливаем шрифт
L:=["Линия 1","Линия 2","Линия 3"]
case SelectDialog("Выберите линию",L,L[1]) of
{
"Okey":
{
GotoRC(1,1)
WWrite(dialog_value) # Печатаем выбранную строку
WDone()
}
"Cancel": {}
}
end
```

Может быть, единственное, что мне не нравится в Icon'e, это семантическая многозначность операционных символов (разрешаемая контекстно). Вот цитата из Cristophera [2, с. 41]:

"The vertical bar, "|", read "«or»", looks like a binary operator but does not behave like one".

Другой пример: символ "\" в выражениях "\x" и "|(z to y)\x" обозначает различные операторы (в первом случае проверяется, имеет ли значение переменная x, а во втором – является ли ограничителем генерируемых значений).

У Icon’a есть сложности с кириллицей. Он «не понимает» кириллические имена файлов при стандартном открытии f:=open(“имя_моего_файла”,”rt”) либо когда имя файла передается в командной строке

```
icon_prog.exe имя_моего_файла
procedure main(args))
f:=open(args[1],”rt”)
…….
end
```

Чтобы обойти эту сложность, следует работать в графическом режиме и обращаться к соответствующему «диалогу».

Большинство проблем при парсировании кириллического текста Вы можете обойти, создав собственное множество символов:

```
allRus:='АБВГДЕЁЖЗИЙКЛМНОПРСТУФХЦЧШЩЬЪЫЭ
ЮЯабвгдеёжзийклмнопрстуфхцчшщьъыэюя'
```

И при необходимости можете создать объеденное множество:

```
allLetters:= &ascii++allRus
……..
```

Теперь несколько слов о математических возможностях Icon’a (которые меня не очень интересуют и в дальнейшем изложении специально не рассматриваются). Язык обладает вполне достаточным набором операторов и функций для того, чтобы быть алгоритмически полным и в «вычислительном» смысле. Он работает с целыми и реальными числами. Имеет специальные средства для манипулирования коплексными и рациональными числами.

Бинарные операторы:

* – умножение

/ – деление

% – остаток (5%2=1)

\+ – сложение

\- – вычитание

Следует помнить, что звездочка (*), используемая как префиксный оператор, возвращает длину объекта.

Информацию о языке, а также дистрибутивы самого языка можно получить по адресу http://www.cs.arizona.edu/icon. Установите Icon (и «пропишите»!). Для написания простых программ (файл с исходным текстом должен иметь расширение.icn) можете пользоваться любым текстовым редактором.

Если у Вас сложный проект, состоящий из нескольких icn-файлов, то предварительно компилируются отдельные части программы:

wicont –c <имя_файла>.icn

Затем для компиляции и получения исполняемого модуля:

wicont <имя_главного_файла>.icn

Главный icn-файл содержит procedure main(…) и декларативы link с именами файлов проекта:

```
link <имя_1>.icn
……
link <имя_N>.icn
……
procedure main(…)
……
end
procedure p1(…)
……
end
procedure pn(…)
……
end
```

Некоторые книги на английском языке доступны в pdf-формате по адресу http://www.cs.arizona.edu/icon/books.htm. Литературные источники по данной теме на русском языке мне не известны, за исключением сотни строк в статье А. Зубинского [2], цитатой из которой я и завершу введение: «Заинтересовавшимся… можно порекомендовать отличную online-версию книги Icon Programming Language Handbook Томаса Кристофера (пригодный к чтению на любых платформах pdf-формат) и сами системы Icon

и Idol (объектно-ориентированное раширение Icon. – О.Ч.), распространяемые в исходных текстах. В любом случае изучение этих языков доставит удовольствие и вооружит вас мощнейшим инструментом как быстрого прототипирования сложных программ, так и "реактивного" создания целого класса очень нужных приложений, для которых любой другой инструментарий не подходит из-за слишком больших трудозатрат».

Учебное издание

О.Г. Чанышев

Основные элементы
языка программирования Icon

Учебное пособие

Технический редактор *Н.В. Москвичёва*

Редактор *Л.Ф. Платоненко*

Подписано в печать 11.11.04. Формат бумаги 60x84 1/16.
Печ. л. 3,4. Уч.-изд. л. 3,1. Тираж 100 экз. Заказ 581.

Издательство Омского государственного университета
644077, г. Омск-77, пр. Мира, 55а, госуниверситет

Электронный архив библиотеки Омского университета